Delivery context negotiated by mobile agents using CC/PP

Rosa Gil, Roberto García, Jaime Delgado

Universitat Pompeu Fabra (UPF), Departament de Tecnologia,
Pg. Circumval·lació 8, E-08003 Barcelona, Spain
{rosa.gil,roberto.garcia,jaime.delgado}@upf.edu

Abstract. Our approach to content negotiation is a framework of mobile agents, where the agents can migrate from user devices to negotiation servers in order to get more resources. We took this approach and now we introduce new features in the architecture. The key idea is content customisation depending on device description with CC/PP (Composite Capability/Preference Profile). The objective is twofold. First, to improve consumer experience adjusting contents to consumption devices. Second, to rationalise network and device use spending only the necessary resources.
Altogether, it is a new step in the direction marked by the use of mobile agents in mobile devices. This way, computation and bandwidth consumption can be moved out of mobile devices to network devices, where these resources are cheaper.
Moreover, in contrast to direct browser-server content negotiation, our agent based negotiation framework provides independence between content negotiation and its consumption, i.e. content can be negotiated and experienced in different user devices, thus better adjusting to user preferences.
All this would be especially relevant when third generation (3G) mobile devices are widely available and more sophisticated multimedia content is available in mobile delivery contexts.

1 Introduction

First of all, we introduce our Mobile Agents Negotiation Framework. The work presented in this paper is build upon this framework in order to manage heterogeneous digital content “delivery contexts”. The next introduction subsections cover the framework and the extensions.

Then, in Sections 2 and 3, we go into detail about how to describe these “delivery contexts” and how to make the negotiation process “context aware”. Finally, in Section 4, the conclusions about current work and future plans on the integration of this framework with other technologies are presented.

1.1 Mobile agents negotiation framework

As we developed in previous paper [1], mobile agents are a key aspect in order to achieve a flexible and robust generic negotiation architecture, where users are represented by agents who can migrate to negotiation servers as we see in **Fig. 1**. Once there, they can negotiate improving messages quantity interchange and computational power.

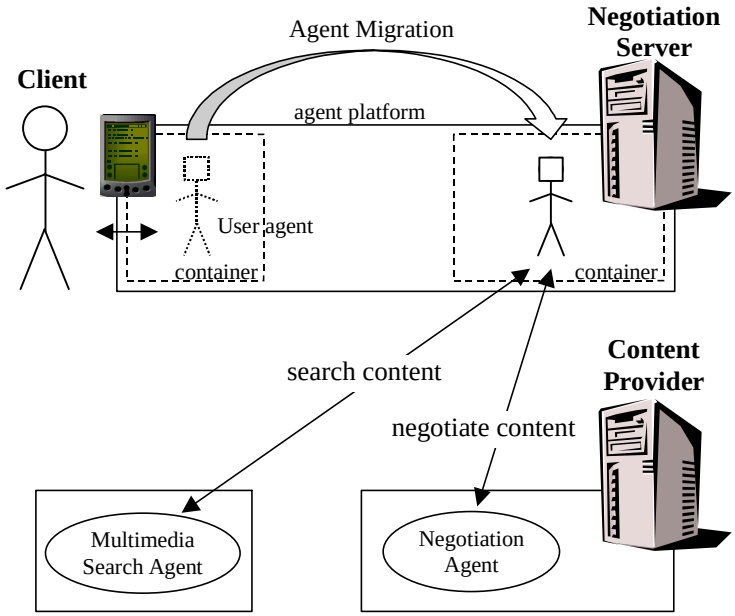

Fig. 1. Negotiation mobile-agent scenario

These ideas have been developed using JADE-LEAP [2] agents and intra-platform mobility. The framework comprises a set of containers and their type adjusts to the hardware platform that gives them support.

There is a main container hosted by the Negotiation Server suited for Java2 SE [3]. For mobile devices, the choices are container for PersonalJava [4] and light containers for Java2 ME and MIDP [5].

Mobility is managed from a rules inference engine implemented with Jess located at the main container. It contains a set of rules that move agents when it is convenient and balance the load of the different containers.

For instance, when a mobile agent wants to start a negotiation, it is moved to the main container at the Negotiation Server. There, the possibly intense negotiation message exchange and reasoning process support can be developed using more appropriate computational and network resources. More details are given in [1].

At the bottom of **Fig. 1**, there are the multimedia specific parts. They are being implemented as part of the NewMARS project [6]. It is a digital content portal with agent negotiation support. Its semantic approach, based on using ontologies like an

Intellectual Property Rights one called IPROnto [7], provides means to enable advanced agent based negotiation of digital contents.

1.2 Improving the architecture

The architecture depicted in the previous section is only the first step towards a mobile and transparent content management platform.

In this section the new features that have been planned are introduced. They are basically necessary because delivery contexts are becoming more and more heterogeneous. However, users do not want to have to deal with different particularised interfaces.

Mobile agents can become this homogeneous content management system and provide the means to make content negotiation and customisation particularities totally transparent to final users.

Moreover, at the same time that delivery contexts diversity increases, new specialised technologies appear. They are related to delivery features but also to other accessory ones with which our mobile agents solution is required to interact. Therefore, we also present our preliminary explorations of an interoperability framework for mobile agents solutions.

More details about all these ideas are presented in the next subsections and summarised in the new scenario presented in **Fig. 2**.

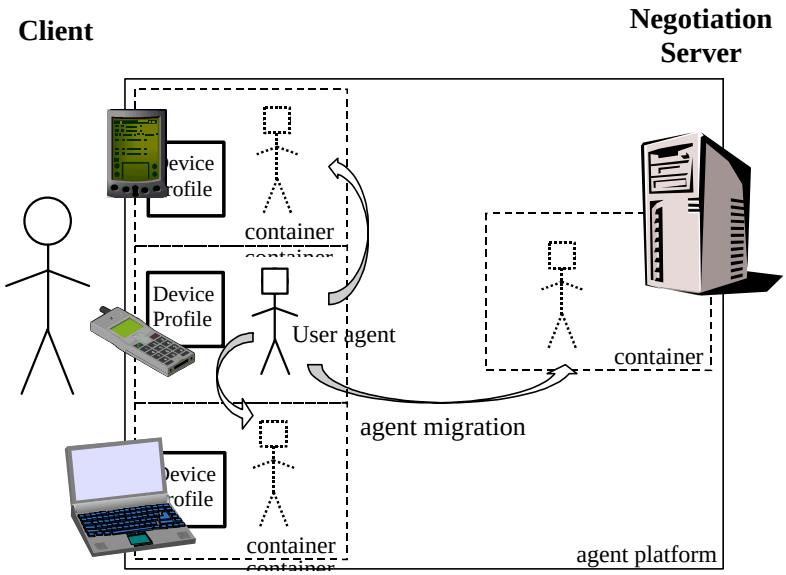

Fig. 2. Negotiation mobile-agent platform supporting multiple user devices

1.2.1 Multiple devices

As we have already pointed out, there is an increasing range of final users devices that cannot be translated to an even more heterogeneous set of interfaces for them. Mobile agent can help users deal with this diversity but, in order to do that, mobile agents must be aware of the kind of devices they must deal with.

Therefore, mobile agents need device descriptions where their characteristics are explicitly stated. There is a W3C [8] initiative that is specifically conceived for this task, the CC/PP [9] one. A more detailed description of CC/PP is presented in Section 2.1.

The mobile agent knows the devices it is in charge of through the corresponding CC/PP profiles. They are explicitly stored in agent's knowledge base or implicitly, by reference. More details about how this is implement in our mobile agents appear in Section 2.3.

1.2.2 Content customisation

Once the mobile agent is aware of the different delivery context it manages for its human user, it must take them into account when it is negotiating content.

The mobile agent starts negotiating with a content provider and informs it about the delivery context of the content. This can be done sending explicitly the content of the corresponding CC/PP profile or a reference of it, eventually augmented by the concrete device particularities.

Using the device profiles, the content provider can adjust its offer to the requested delivery context. On one hand, it can adjust context characteristics, for instance image size, colour depth, streaming bandwidth, etc. On the other hand, it can adjust economical and utilisation conditions accordingly.

Besides that, the existing negotiation protocol stays unaffected. The user mobile agent can directly accept the proposed conditions or start a negotiation cycle that approaches them to those it thinks its user may require.

A detailed view of how customisation is managed and how device profiles are used during negotiations is shown in Section 3.

1.2.3 Integrating with future

We are looking at future tendencies. An important task will be to integrate mobile agents platforms with other initiatives. For instance, P2P, Web Services, Grid… The details are discussed in section 4.

2 Describing delivery contexts

A delivery context could be defined as a set of attributes that characterises the capabilities of the access mechanism and the preferences of the user. An access mechanism is a combination of hardware (including one or more devices and network connections) and software (including one or more user-agents) that allows a user to perceive and interact with the Web using one or more interaction modalities (sight, sound, keyboard, voice, etc.). Web servers and applications that adapt content for

multiple access mechanisms must be sensitive to delivery context. Composite Capability/Preference Profile (CC/PP) corresponds to the W3C initiative in this line.

2.1 CC/PP Composite Capability/Preference Profile

CC/PP Composite Capabilities / Preferences Profile (CC/PP) provides a standard way for devices to transmit their capabilities and user preferences. It was originally designed to be used when a device requests web content via a browser so that servers and proxies can customize content to the target device, i.e. support device independence. It is a proposed industry standard for describing delivery context.

Moreover, the World Wide Web Consortium (W3C) hosts two activities that are relevant to this specification. The first of these is the Device Independence Working Group, which is part of the W3C Interaction domain. This group has produced a document describing device independence principles [10] and two informal drafts, one discussing delivery context [11], i.e. mechanisms like CC/PP, and the other one discussing authoring [12].

The second relevant activity hosted by the W3C is the CC/PP Working Group, which is also part of the W3C Interaction domain [9].

UAProf [13], User Agent Profile, is a concrete implementation of CC/PP developed by the Open Mobile Alliance (OMA) that like the W3C is organised into areas or groups and UAProf is part of the Mobile Application Group. UAProf is an implementation of CC/PP aimed at WAP-enabled [14] mobile terminals.

CC/PP profiles contain capability and preference information sent from a client to a server. In order to validate CC/PP profiles, there must be a set of rules that determine what constitutes a valid profile. According to the CC/PP structure and Vocabularies Working Draft a CC/PP profile must first meet the constraint of being a valid XML and Resource Description Framework (RDF) document [15]. The W3C's RDF validation service [16] can be used to validate a profile in this way.

Currently there are two protocols developed for CC/PP exchange based on HTTP, CC/PP-ex [17] and W-HTTP, the last one based on WAP [18]. These protocols have many common features. They send CC/PP information in two forms: references profiles and profile diffs. A reference profile is sent as a URI between the client and the server. The server then uses this URI to retrieve the profile from a third source known as a profile repository. Profile diffs on the other hand are sent as in-line XML fragments and may or may not be present in the headers. Profile diffs are associated with a sequence number that indicates processing order.

2.2 Processing CC/PP

A first step in processing CC/PP is to make the current generation presentation-oriented Web technology interoperable with the next-generation Semantic Web technology [19]. For example, CSS [20] style sheets are currently not able to take CC/PP profiles into account. CSS has, however, a feature that is closely related to CC/PP, and allows the specification of device dependent style rules. **Fig. 3** shows an

example of a style sheet that uses smaller fonts on mobile devices screens than desktops screens for the same document.

```
@media screen { min-width:32px
body { font-size: 8 pt }
}
@media screen { min-width:640px
body{ font-size: 12pt}
}
```

Fig. 3 Device dependent style rules as CSS3 extension.

Style engines need to be able to deal with these features in order to take full advantage of the information specified in CC/PP delivery contexts.

Note that the need to take CC/PP information into account also applies to XSLT [21] transformation engines. One could, for example, imagine an extension of XSLT's mode concept. For example, transformation rules could be selected in a way similar to that of the media rules in CSS. In such a hypothetical extension, see **Fig. 4**, one could, for instance, define a rule for creating a two column layout only if the output medium is a desktop and the screen is wider than 1024 pixels.

```
<xsl:templatematch="body"
mode="screen and(min-width:1024px)">
...
<fo:region-bodycolumn-count="2"/>
...
</xsl:template>
```

Fig. 4. Using XSLT transformation engines

In addition to taking information about delivery contexts into account, style sheets also need to take into account the semantic information that is contained in the metadata associated with the content. Currently, style selector mechanisms only match on the syntactic properties of the underlying XML document hierarchy. This applies both to the selector mechanism used by CSS and to the XPath [22] selectors used by XSLT.

In all examples above, the rules were intended to match on the <body> element of an HTML document. However, to use the current generation CSS and XSLT engines to process general metadata is not practical to match on the semantic properties of metadata. For CSS and XSLT processors, RDF is just XML. As a result, it is very hard to write, for example, a rule that matches on all alternative XML serializations that are allowed for RDF [15].

A more serious problem, however, is that it is impossible to write CSS or XSLT rules that make use of the semantic features of RDF Schema [23] (RDFS). For instance, a style rule that applies to all objects that are instances of a specific RDFS class.

Future semantics-aware selector mechanisms would allow specification of rules in terms of the RDF semantics expressed in the metadata. This would extend the currently used CSS and XPath selectors, which are based on the XML syntax

encoding the semantics. Consider the extended XSLT example rule in **Fig. 5**, which uses the RDF-aware query language RQL [24] for its selector, instead of XPath.

```
<xsl:template match="RQL(http://dmag.upf.es/schema.rdf#VideoMatrix)">
...
</xsl:template>
```

Fig. 5 Semantic matching of XSLT rules using RQL selectors

It matches on all resources that are instances of (subclasses of) the RDF class VideoMatrix. Given the fact that our RDF Schema would define “Matrix Reloaded” as a subclass of VideoMatrix, the rule would also match on the other HTML fragments. Such rules that employ the semantic relations defined in the metadata are currently impossible to write in XSLT.

2.3 Using CC/PP

Some API’s as DELI (A Delivery context Library for CC/PP and UAProf) and Intel ® CC/PP SDK [25] have been implemented in the latest times to solve some of the previous problems and content negotiation has been implemented using CC/PP and WAP UAProf [26]. However, there is not a standard way of doing all this, but it is under development [27].

Meanwhile, we have preferred to use the available mobile agents communication facilities, i.e. FIPA-ACL messages. The mobile agent knows the devices it is in charge of through the corresponding CC/PP profiles. They are explicitly stored in agent’s knowledge base or implicitly by references, URL, that point to the WWW location where they are stored. Moreover, there can be particular modifications to these profiles, for instance a device with an upgraded amount of memory.

In the next section, it is shown how agents interchange CC/PP profiles and use them during negotiation.

3 Negotiation with customisation

Now, we go into the implementation part. First, in Section 3.1, the negotiation process implemented by mobile agents is presented. This process is extended, in Section 3.2, with CC/PP profile exchange and used during the negotiation process.

3.1 Negotiation process

In the beginning, the mobile agent resides at one of the user devices, which is mobile agents capable, i.e. it has a mobile agents container installed. The user interacts through the agent’s GUI to determine all the criteria required to select the content he is interested in. Then, the agent enters the search phase and it migrates to another agent-platform container. This new container is one of the server kind, where it has

better Internet connectivity and processing power. Thanks to these greater resources, it can carry on less constrained searches and a more accurate negotiation process of the required content.

When the user agent owns enough information, it can start the automatic negotiation process through a protocol defined by rules. The retrieved licensing agent, which is in charge of negotiating the license of the desired content, is contacted and a call for proposals is issued. An initial offer is received, if the licensing agent really has the requested content. From this point, some counter-offers may be interchanged till the negotiation ends due to a reject or an agreement.

The negotiation results are then communicated to the user. To facilitate user interaction, the user agent returns to the agent-platform container at the users mobile device.

3.2 Negotiating customisation

The first change to allow negotiating customisation is to make the content provider agent aware of the delivery context conditions. ACL messages with the "inform" communicative act are used for this [28]. An example is shown in **Table 1**.

More specifically, the "inform" communicative act is used to communicate explicit delivery context information, i.e. FIPA-ACL message content is the CC/PP profile serialised in its RDF/XML form [29]. To communicate it implicitly, the "inform-ref" communicative act is used and the ACL message content is the URL reference pointing to the CC/PP online version. Although the content is in RDF/XML form, we have also inspired ourselves on the FIPA device ontology [30].

The use of RDF/XML profile encoding allows a direct integration of all this information in the negotiation process. DAMLJessKB [31] allows adding and extracting facts from Jess engine. In other words, RDF (including CC/PP delivery context profiles) is incorporated into the Jess engine.

Then, the negotiation policies are implemented by Jess rules, which can be exported to a common rules interchange format, RuleML[32], and also imported. The rules take into account delivery context information to steer the negotiation. For an example see **Table 2**.

Table 1. Delivery context implicit exchange by an inform-ref preformative ACL message, example fragment extracted from the Siemens S55 mobile phone CC/PP profile available at http://communication-market.siemens.de/UAProf/S55_00.xml

```
(inform
   :sender User1MA
   :receiver ContentProviderA
   :content (<?xml version="1.0" encoding="UTF-8" ?>
      <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:prf="http://www.wapforum.org/profiles/UAPROF/ccppschema-20010330#"
           xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
       <rdf:Description ID="S55_Profile">
             <prf:component>
                <rdf:Description ID="HardwarePlatform"> …
             <prf:component>…
      </rdf:Description>
```

```
      </rdf:RDF>)
   :language fipa-rdf0 )
```

Table 2. Jess rule that uses RDF metadata imported by DAMLJessKB to detect image support

```
(defrule are-images-supported
   (PropertyValue ?resource
      http://www.w3.org/1999/02/22-rdf-syntax-ns#type
      http://www.wapforum.org/…/UAPROF/ccppschema-20010330#HardwarePlatform)
   (PropertyValue ?resource
      http://www.wapforum.org/…/UAPROF/ccppschema-20010330#imageCapable "Yes")
=>
   (assert (include-images)))
```

4 Future work

In this final section we present some future plans that try to solve the problems we have found when trying to integrate different agent technologies.

4.1 Mobile Agents integration

During the development of the Mobile Agents Negotiation Framework, we have found some tasks that are not well faced by mobile agents solutions. For instance, negotiation decision tasks that are computation intensive. We have considered more convenient to encapsulate all these intensive computations as independent services and implemented by other technologies, for instance Grid networks [33].

At this point we have to face interoperability problems between this two different kinds of mobile computation networks. The first one, which we have used till now, seems more appropriate for user mobility environments as the one resolved by the previous content negotiation architecture. The second one, the Grid, appears as the best choice in the "server" side. When intensive computations are required the best choice is to transparently integrate many computational resource into a Grid. Inside this Grid, the required computations can be distributed to attain the best affordable computational throughput.

Therefore, as we have seen that combining both mobile code solutions is the best option, an interoperability layer is necessary. In Section 4.1.1, we present our preliminary integration architecture.

We propose to use a Peer-to-Peer solution in order to transparently and dynamically integrate both. Peer groups are configured dynamically from global UDDI repositories using Web Services tools. Once the required services have been found and configured, the ad-hoc peer group is established.

4.1.1 Peer-to-Peer interoperability layer

Our new proposed architecture model, presented in **Fig. 6,** shows how the different technological pieces are combined to meet the requirements. These pieces are Grid

Computing, P2P, Semantic Web Services and Agents. These technologies are organized into a layer cake design.

At the bottom layer, there are the computational and storage resources managed by the Grid. This layer contains a set of grids that conform resource-sharing spaces with a unique logical access point. The problem is that users see these spaces like isolated islands, and, altogether, they behave like static groups of resources that must be put together by hand.

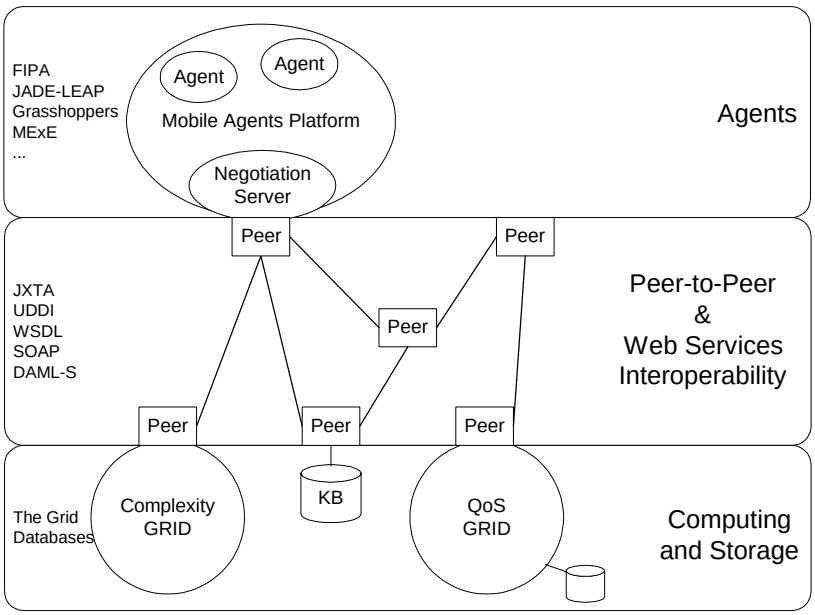

Fig. 6 Architecture model

Peer to peer technologies are the medium that enables an open communication among totally distributed resources. When P2P technologies are deployed over the Internet, a potentially global communication space is obtained. Therefore, P2P has been stacked over the Grid layer, more concretely the JXTA[34] package.

JXTA provides the building blocks to construct a network of peers over the Internet infrastructure using Web technologies. Each peer sets an access point to the resources it manages and, at the same time, is the medium to reach other peers. This feature has been used and a peer point has been associated to logical Grid access points.

5 Conclusions

A great variety of mobile devices are spreading over the world, but they are not the only ones, they share communications environment with desktops and laptops among

other devices. An end user can have many of them; customisation is the only way to get desired content almost everywhere. So if we want to design a true service negotiating multimedia content, customisation is a goal to achieve. CC/PP seems to be a veritable tool and we have reviewed how it could be useful.

However, there is still a lot of work to do in order to integrate the semantic capabilities of CC/PP in the customisation process. We have outlined a server side specific option based on the use of the Jess rule engine with RDF semantics capabilities. However, in order to see a spread adoption of CC/PP, semantic capabilities should be integrated in style sheet technologies, i.e. CSS or XSLT.

Moreover, thinking about the nearest future it is important to have in mind a global idea about the network, because customisation implies to be capable of working in many technologies, as P2P or Grid presented in Section 4. Thus, we want to propose a complete business solution based on mobile agents. They can travel around Mobile Agent Platforms and negotiate in our name.

References

1. Delgado, J.; Gallego, I.; García, R. and Gil, R.: An Architecture for Negotiation with Mobile Agents. Mobile Agents for Telecommunication Applications, MATA 2002. Lecture Notes in Computer Science, Vol. 2521. Springer-Verlarg (2002) 21-31
2. JADE-LEAP, http://leap.crm-paris.com
3. Java 2 Platform Standard Edition, http://java.sun.com/j2se
4. Pjava, Personal Java. http://java.sun.com/products/personaljava/pj-emulation.html
5. Mobile Information Device Profile, http://java.sun.com/products/midp
6. AREA 2000 Spanish Government Research Project (TIC2000-0317-P4-05), NewMARS subproject, http://dmag.upf.es/newmars
7. Delgado, J., Gallego, I., Garcia, R., Gil, R.: An ontology for Intellectual Property Rights: IPROnto. Extended poster abstract, International Semantic Web Conference (2002) http://dmag.upf.es/papers/ExtAbstractPosterISWC.pdf
8. World Wide Web Consortium, http://www.w3c.org
9. Composite Capabilities/Preferences Profile Working Group, http://www.w3.org/Mobile/CCPP
10. Device independence principles, http://www.w3.org/TR/2001/WD-di-princ-20010918
11. Discussing delivery context, http://www.w3.org/TR/2002/WD-di-dco-20021213
12. Discussing authoring, http://www.w3.org/TR/2002/WD-acdi-20021018
13. DELI: A Delivery context Library for CC/PP and UAProf. http://www-uk.hpl.hp.com/people/marbut/DeliUserGuideWEB.htm
14. WAP forum. http://www.wapforum.org
15. Lassila, O., Swick, R.R. (editors): Resource Description Framework (RDF), Model and Syntax Specification. W3C Recommendation 22 February 1999 http://www.w3.org/TR/REC-rdf-syntax
16. RDF validation service. http://delicon.sourceforge.net/
17. Ohto ,H. and Hjelm, J.: CC/PP exchange protocol based on HTTP Extension Framework. W3C Note 24 June 1999, http://www.w3.org/TR/NOTE-CCPPexchange
18. WAP Forum Releases, Technical Specification, http://www.wapforum.org/what/technical.htm
19. Garcia, R.; Delgado, J.: Brokerage of Intellectual Property Rights in the Semantic Web. 1st Semantic Web Working Symposium (SWWS-1), Stanford, CA (2001)

20. Wugofski, T., Dominiak, D., Stark, P. and Roy, T.: CSS Mobile Profile 1.0. W3C Candidate Recommendation 25 July 2002, http://www.w3.org/TR/css-mobile
21. Clark, J. (ed.): XSL Transformations (XSLT) Version 1.0 W3C Recommendation 16 November 1999, http://www.w3.org/TR/xslt
22. XML Path Language, http://www.w3.org/TR/xpath
23. Brickley, D. and Guha, R.V. (eds): RDF Vocabulary Description Language 1.0: RDF Schema. W3C Working Draft 30 April 2002. http://www.w3.org/TR/rdf-schema
24. Karvounarakis, G.; Alexaki, S.; Christophides, V.; Plexousakis, D.; Scholl, M.: RQL: A Declarative Query Language for RDF. The Eleventh International World Wide Web Conference, 2002
25. M.Bowman, R.D. Chandler, D V. Keskar. Intel Labs. Delivering Customized Content to Mobile Device Using CC/PP and the Intel ®CC/PP SDK.
26. Butler M. H.: Implementing Content Negotiation using CC/PP and WAP UAProf. Hewlett-Packard (2001)
27. Composite Capability/Preference Profile (CC/PP) Processing Specification. Public Draft, version 0.5 (2003)
28. FIPA Communicative Act Library Specification, http://www.fipa.org/specs/fipa00037/SC00037J.html
29. Adapted Content Delivery for Different Contexts http://opera.inrialpes.fr/people/Tayeb.Lemlouma/Papers/saint2003.pdf
30. FIPA Device Ontology Specification. http://www.fipa.org/specs/fipa00091/XC00091C.pdf
31. DAMLJessKB documentation. http://edge.mcs.drexel.edu/assemblies/software/damljesskb/javadocs/edu/drexel/gicl/daml/damljesskb/DAMLJessKB.html
32. RuleML http://www.semanticweb.org/SWWS/program/full/paper20.pdf
33. Foster, I.; Kesselman, C.; Tuecke, S.: The Anatomy of the Grid, Enabling Scalable Virtual Organizations. International J. Supercomputer Applications, 15:3 (2001)
34. JXTA project, http://www.jxta.org